Remarks

I. The Amendment to the Claims

Claims 1, 10, 14-17, 30 and 37 have been amended. Claim 24 has been canceled.

II. 35 U.S.C. §112

Claims 4-7, 10-1 6, 22, 24-27, 31 -33, 35-37, 39-40 stand rejected under 35 U.S.C. §112, first paragraph, as allegedly failing to comply with the enablement requirement. For this rejection, the Office Action states that reference will be made to the Request to Invoke Interference, where Applicant maps the claim limitations to portions of Applicant's provisional application 60/098,296.

A. Claims 4 and 5

Regarding claims 4 and 5, the Office Action states:

> Claims 4-5 recite limitations regarding the receipt of a code associated with the packet in which the code indicates that the data portion is either reassembleable or not reassembleable. The mapped portions in the provisional do not provide the proper support for this subject matter. The portions do not disclose any actual "receipt" of a code nor do they disclose any indication as to whether the data is reassembleable or not. Summarizing a header's type into a single status word does not appear to equate to receiving a code that indicates that the data packet is reassembleable or not. Indicating that the header specifies a fast-path connection does not appear to specify if the data is reassemblable or not. The portions mapped also do not specify the type of data storage area to be a re-assembly data storage area or a non-re-assembly data storage area. The portions merely indicate small and large buffers and has nothing to do with whether they are re-assembly or non-re-assembly data storage areas.

Applicants respectfully disagree with this rejection. According to U.S. Patent No. 6,480,049 to Muller et al. ("Muller"), whether data from a packet is "re-assembleable" refers to whether "the packet was formatted with one of the set of predetermined protocols." Abstact. If so "its data is re-assembled in a re-assembly buffer with data from other packets in the same communication flow." *Id.* "The network interface may be configured to re-assemble only packets that are formatted in accordance with one or more of a set of pre-selected protocols. For example, where the network from which the

packet is received is the Internet, the network interface may be configured for packets adhering to the Internet Protocol and the Transport Control Protocol. A header parser module of the network interface may examine a header portion of a packet to determine if it is compatible with (e.g., reflects) the pre-selected protocols." Column 4, lines 36-45.

Applicants' provisional application 60/098,296 ("the '296 app.") says much the same thing with different words. It is directed to the same protocol suite (TCP/IP), has a network interface that receives a packet for which the "header is fully parsed by hardware and its type is summarized in a single status word." Column 6, lines 36-38. "If the type field contains our custom INIC type (TCP for example): A. If the header buffer specifies a fast-path connection, allocate one or more mbufs headers to map the header and possibly data buffers." Page 41, lines 25-27. The "data buffers" can be used to reassemble data from the received packets that belong to the "fast-path" communication flow. "As mentioned above, the fast-path flow puts a header into a header buffer that is then forwarded to the host. The host uses the header to determine what further data is following, allocates the necessary host buffers, and these are passed back to the INIC via a command to the INIC. The INIC then fills these buffers from data it was accumulating on the card and notifies the host by sending a response to the command." Page 13, lines 17-21. On the other hand, as noted in the '296 app. at page 85, lines 26-30 and page 87, lines 9-15, if the "frame status" classifies the packet as "slow-path," it may be moved into a small host buffer where it is not reassembled with any other packets, but rather processed individually by the host CPU.

Regarding the Office Action's assertion of "actual 'receipt' of a code," note that Muller does not disclose receiving the code from a network like receiving a packet from a network, and claims 4 and 5 also do not recite "receiving a code from a network." Indeed, although the phrase "receiving a code" is found in claims 4, 5, 6 and 7, it is not found anywhere in the specification of Muller. Instead, Muller discloses that one part of the "network interface" creates a "code" that is then used by another part of the "network interface." As stated in column 4, lines 34-35 of Muller: "The operation code may be generated or assigned by module that maintains the flow database."

For all the above reasons, applicants respectfully submit that it is clear that claims 4 and 5 are enabled by the '296 app., which was incorporated by reference in each application in the chain leading up to and including the present application.

B. Claim 6

Regarding claim 6, the Office Action states:

> Regarding claim 6, the claim requires receiving a code with the first packet, the code indicting that the first packet does not contain a data portion. The mapped portions do not appear to disclose this subject matter. A field in a header buffer does not equate to the actual receipt of a code with the packet. The rest of the portions mapped discuss the general examining of a frame header to generate an event from it and has no relevance to whether a code received with the packet indicates that the packet does not contain a data portion.

Applicants respectfully disagree with this rejection. Claim 6 recites: "The method of claim 1, further comprising receiving a code with said first packet, said code indicating that said first packet does not contain a data portion." Applicants respectfully assert that "receiving a code with said first packet" does not necessarily mean "receiving a code from the network with said first packet," as apparently construed by the Office Action. In accordance with this is Muller, which does not disclose "receiving a code from the network with a packet." The field in the header buffer that indicates whether a packet has valid data ('296 app. page 14, lines 6-7) can be considered to be a code that is received by the host with the packet that is in the header buffer.

For at least these reasons, applicants respectfully submit that claim 6 is enabled by the '296 app.

C. Claim 7

Regarding claim 7, the Office Action states:

> Regarding claim 7, the claim requires receiving a code with the second packet that indicates that the packet is smaller than said predetermined size. It appears from Applicant's mappings that there is no code provided and the relied upon "length" is simply determined based on the data of the packet that it holds.

Applicants respectfully disagree with this rejection. Claim 7 recites: "The method of claim 1, further comprising receiving a code with said second packet, said code indicating that said second packet is smaller than said predetermined size." Applicants respectfully assert that "receiving a code with said second packet" does not necessarily mean "receiving a code from the network with said second packet," as apparently construed by the Office Action. In accordance with this is Muller, which does not disclose "receiving a code from the network with a packet." The information contained in the header buffer about the data, such as the length ('296 app. page 70, lines 12-13) can be considered to be a code that is received by the host with the packet that is in the header buffer, which indicates whether the packet is smaller than a predetermined size.

For at least these reasons, applicants respectfully submit that claim 7 is enabled by the '296 app.

D. Claim 10

Regarding claim 10, the Office Action states:

> Regarding claim 10, the claim requires that association of a code with the packet to indicate whether the data portion of the packet is re-assembleable with a data portion of another packet in said communication flow. The mapped portions only describe the categorization of a packet to a CCB, but nothing regarding whether the data of the packet is reasembleable with the data from another packet. Also regarding claim 10, the claim requires the receipt of a set of descriptors from the host, wherein the aggregate size of the descriptors approximates a cache line size of the host computer. The mapped portions do not specify the receipt of descriptors, let alone wherein the aggregate size of the descriptors approximates a cache line size of the host computer. Rather the mapped portions relate to the cache line size that can be accommodated, and it appears the Pmi performs bursts until is has aligned the transfers, which appear to be the opposite of receiving descriptors that approximates the size. Claim 10 also recites a re-assembly storage area. The portions mapped also do not specify the type of data storage area to be a re-assembly data storage.

Applicants respectfully disagree with this rejection. Regarding the Office Action assertion that "The mapped portions only describe the categorization of a packet to a CCB, but nothing regarding whether the data of the packet is reasembleable with the data from another packet," as noted above regarding claims 4 and 5, the '296 app. discloses

the reassembly of data from packets that correspond to a "fast-path" communication flow. Such a "fast-path" flow is taught to be possible when the packets correspond to a CCB controlled by the network interface. As stated on page 41, lines 25-27 of the '296 app.: "If the type field contains our custom INIC type (TCP for example): A. If the header buffer specifies a fast-path connection, allocate one or more mbufs headers to map the header and possibly data buffers." The "data buffers" can be used to reassemble data from the received packets that belong to the "fast-path" communication flow. As stated on page 13, lines 17-21, "the fast-path flow puts a header into a header buffer that is then forwarded to the host. The host uses the header to determine what further data is following, allocates the necessary host buffers, and these are passed back to the INIC via a command to the INIC. The INIC then fills these buffers from data it was accumulating on the card and notifies the host by sending a response to the command."

Regarding the Office Action assertion that "The mapped portions do not specify the receipt of descriptors, let alone wherein the aggregate size of the descriptors approximates a cache line size of the host computer," applicants direct the Examiner's attention to the section on page 14 of the '296 app. entitled "3.2.2 Receive Interface Details," which describes sets of descriptors being passed from the host to the INIC.

Regarding the Office Action assertion that "The portions mapped also do not specify the type of data storage area to be a re-assembly data storage," applicants note that, similar to the discussion above regarding claims 4 and 5, the "data buffers" can be considered to be a "re-assembly storage area" for data from packets corresponding to the "fast-path" flow.

For at least these reasons, applicants respectfully submit that claim 10 is enabled by the '296 app.

E. Claims 11 and 12

Regarding claims 11 and 12, the Office Action states:

> Claims 11-12 recites a re-assembly storage area. The portions mapped do not specify the type of data storage area to be a re-assembly data storage

Applicants respectfully disagree. "Large buffers" or "data buffers" can be considered to be re-assembly buffers, and are disclosed as being substantially equal in size to one memory page of said host computer and only storing the data portions of packets in the "fast-path" flow.

For at least these reasons, applicants respectfully submit that claims 11 and 12 are enabled by the '296 app.

F. Claim 14

Regarding claim 14, the Office Action states:

> the mapped portions do not indicate how said first storage area identifier is identifiable by an index in said data structure. The mapped portions do not provide an index. The mapped portions recite adding the data buffer handle and the data buffer address to a queue, and that two values are extracted each time at dequeuing, but there doesn't appear to be an index used in order to identify the first storage area identifier in a data structure.

Applicants respectfully disagree, but have amended claim 14 to recite in part "wherein said first storage area identifier is identifiable by a descriptor in said data structure." Such a descriptor is disclosed, for example, on page 14, lines 27-36 of the '296 app.

For at least this reason, applicants respectfully submit that claim 14 is enabled by the '296 app.

G. Claim 15

Regarding claim 15, the Office Action states:

> Regarding claim 15, the mapped portions do not indicate how said first storage area identifier is identifiable by an index in said data structure. The mapped portions do not provide an index. The mapped portions recite adding the data buffer handle and the data buffer address to a queue, and that two values are extracted each time at dequeuing, but there doesn't appear to be an index used in order to identify the first storage area identifier in a data structure.

Applicants respectfully disagree, but have amended claim 15 to recite in part "using said descriptor to identify said first storage area for storing a portion of said first

packet." Such a descriptor is disclosed, for example, on page 14, lines 27-36 and page 21, lines 33-41 of the '296 app.

For at least this reason, applicants respectfully submit that claim 15 is enabled by the '296 app.

H. Claim 16

Regarding claim 16, the Office Action states:

> Regarding claim 16, the claim requires configuring a descriptor to store said index of said first storage area identifier to inform said host computer of the use of the storage area. The mapped portions in the provisional do not provide the proper support for this subject matter. It appears that the mapped portions relate to the lNlC maintaining a queue and adding to the queue when the host writes to the header buffer address registers, which appears to not equate to informing the host computer of the use of said first storage area. Rather it appears that the lNlC simply keeps track of when the host computer writes to one of the header buffer address registers.

Applicants respectfully disagree, but have amended claim 16 to recite in part "configuring a queue to store said descriptor of said first storage area identifier to inform said host computer of the use of said first storage area to store one of said header portion and said data portion.." Such a queue to store said descriptor is disclosed, for example, on page 14, lines 27-36 of the '296 app.

For at least this reason, applicants respectfully submit that claim 16 is enabled by the '296 app.

I. Claim 22

Regarding claim 22, the Office Action states:

> Regarding claim 22, the claim requires the receipt of a virtual connection identifier wherein a communication flow can be identified by the virtual connection identifier. The mapped portions in the provisional do not provide the proper support for this subject matter, as there is no mention of a virtual connection identifier in order to identify the communication flow.

Applicants respectfully disagree. The "context number" disclosed in the mapped portions can be considered to be a "virtual connection identifier."

For at least this reason, applicants respectfully submit that claim 22 is enabled by the '296 app.

J. Claim 24

Claim 24 has been canceled.

K. Claim 26

Regarding claim 26, the Office Action states:

> Claim 26 recites an identifier of a location in said re-assembly storage area. The portions mapped do not specify the type of data storage area to be a re-assembly data storage are.

Applicants respectfully disagree. As noted above regarding claims 4 and 5, the '296 app. discloses the reassembly of data from packets that correspond to a "fast-path" communication flow in "data buffers." As stated on page 13, lines 17-21, "the fast-path flow puts a header into a header buffer that is then forwarded to the host. The host uses the header to determine what further data is following, allocates the necessary host buffers, and these are passed back to the INIC via a command to the INIC. The INIC then fills these buffers from data it was accumulating on the card and notifies the host by sending a response to the command."

For at least this reasons, applicants respectfully submit that claim 26 is enabled by the '296 app.

L. Claims 27, 31 and 35-37

Regarding claims 27, 31 and 35-37, the Office Action states:

> Claim 27, 31, 35-37 include the same issues in the above claims.

Applicants respectfully disagree for at least the reasons mentioned above, and respectfully submit that claims 27, 31 and 35-37 are enabled by the '296 app.

III. Priority

The Office Action asserts that applicants have not complied with one or more conditions for receiving the benefit of an earlier filing date under 35 U.S.C. 119 or 120, stating:

> The later-filed application must be an application for a patent for an invention which is also disclosed in the prior application (the parent or original nonprovisional application or provisional application). The disclosure of the invention in the parent application and in the later-filed application must be sufficient to comply with the requirements of the first paragraph of 35 U.S.C. 112. See *Transco Products, Inc. v. Performance Contracting, Inc.*, 38 F.3d 551, 32 USPQ2d 1077 (Fed. Cir. 1994).
>
> The disclosure of the prior-filed application, Application No. 10/005,536 (or any of the other prior applications for which benefit is claimed under 35 U.S.C. 119 or 120) fails to provide adequate support or enablement in the manner provided by the first paragraph of 35 U.S.C. 112 for one or more claims of this application.
>
> Applicant's prior applications do not provide adequate disclosure for the subject matter pertaining to the "hybrid storage area" or the "re-assembly storage area" as claimed (i.e. all limitations regarding these storage areas as well as the claimed functionality).
>
> MPEP 201.07 recites, "The disclosure presented in the continuation must be the same as that of the original application." It appears that the disclosure for this specification contains subject matter not contained in the prior applications.
>
> Accordingly, claims 4-7, 10-12, 14-16, 22, 24, 26-27, 31, 35-37 are not entitled to the benefit of the prior applications.

Applicants respectfully disagree with the Examiner's denial of the benefit of prior applications for claims 1-40. Initially, applicants note that the case relied upon by the Examiner, *Transco*, involved the issue of whether the "best mode" needs to be updated upon the filing of a "continuing application," not whether an application was entitled to the benefit of the filing date of a prior-filed application. As to the latter issue, *dicta* in Transco merely requires "common subject matter" for such entitlement. In general, "the test for sufficiency of support in a parent application is whether the disclosure of the application relied upon 'reasonably conveys to the artisan that the inventor had possession at that time of the later claimed subject matter.' *In re Kaslow*, 707 F.2d 1366, 1375, 217 U.S.P.Q. (BNA) 1089, 1096 (Fed. Cir. 1983)." *Ralston Purina Co. v. Far-Mar-Co.*, 772 F.2d 1570, 1575 (Fed. Cir. 1985). Moreover, cases such as *KangaROOS, U.S.A., Inc. v. Caldor, Inc.*, 778 F.2d 1571, 1574 (Fed. Cir. 1985) hold that a utility patent

may claim priority from a design patent, even though no claims existed in the design patent other than claiming the drawings that were shown. Similarly, cases such as *In re Hogan,* 559 F.2d 595, 608 (C.C.P.A. 1977) hold that "claimed subject matter need not be described *in haec verba* in the application to satisfy the written-description-of-the-invention requirement."

Indeed, *Hogan* involved a situation where a broad claim ostensibly covered later enabled species even though those species were not enabled in the original application. Although Hogan involved chemical arts, which have been held to be less predictable than mechanical or electrical arts, Hogan held that failure to enable those later species did not undermine the claim to the filing date the earlier application. *Hogan* at 606. "To now say that appellants should have disclosed in 1953 the amorphous form which on this record did not exist until 1962, would be to impose an impossible burden on inventors and thus on the patent system. There cannot, in an effective patent system, be such a burden placed on the right to broad claims." *Id.*

Applicants therefore respectfully disagree with the Office Action assertion that "prior applications do not provide adequate disclosure for the subject matter pertaining to the 'hybrid storage area' or the 're-assembly storage area' as claimed (i.e. all limitations regarding these storage areas as well as the claimed functionality)." While it is true that applicants did not use the exact *nonce words* "hybrid storage area" or "re-assembly storage area" that were coined in Muller, it is clear that applicants' parent cases disclosed the subject matter that those words represent.

Perhaps because the nonce words coined by Muller are not terms of art, column 58, lines 8-11 of that reference explains: "A buffer used to store portions of more than one type of packet-such as a header buffer used to store headers and small packets, or a non-re-assembly buffer used to store MTU and jumbo packets-may be termed a 'hybrid' buffer." Similarly, column 4, lines 48-50 of that reference states: "A re-assembly buffer may be used to re-assemble data from multiple packets of a single communication flow." In concert with this explanation of those nonce words is Muller's claim 37, which recites in part: "a re-assembly storage area configured to store data portions of a plurality of packets from a single communication flow;" and "a hybrid storage area configured to

store: header portions of the plurality of packets; and one or more packets smaller than a first predetermined size."

Ample support for these limitations can be found in applicants' early disclosures. For example, regarding the limitation of "a hybrid storage area," that is "configured to store: header portions of the plurality of packets; and one or more packets smaller than a first predetermined size," applicants reproduce below pertinent portions of pages 9 – 13 of the '296 app., with added emphasis and parenthetical insertion of explanatory or nonce words. Support for the limitation of a "re-assembly storage area" that is "configured to store data portions of a plurality of packets from a single communication flow;" can also be found in these portions of the '296 app.

> **2.3.2. Support small and large buffers on receive**
>
> In order to reduce further the number of writes to the INIC, and to reduce the amount of memory being used by the host, we support two different buffer sizes. A small ***(hybrid)*** buffer contains roughly 200 bytes of data payload, as well as extra fields containing status about the received data bringing the total size to 256 bytes. We can therefore pass 16 of these small buffers at a time to the INIC. Large ***(data or re-assembly)*** buffers are 2k in size. ***They are used to contain any fast or slow-path data that does not fit in a small buffer.*** Note that when we have a large fast-path receive, a small ***(hybrid)*** buffer will be used to indicate a small piece ***(header portion)*** of the data, while ***the remainder of the data*** will be DMA'd directly into ***memory (data buffers or re-assembly storage area)***.
>
> ...
>
> **2.4.1. Fast-path 56k NetBIOS session message**
>
> Let's say a 56k NetBIOS session message is received on the INIC. The first segment will contain the NetBIOS header, which contains the total NetBIOS length. A small chunk ***(header portion)*** of this first segment is provided to the host by filling in a small ***(hybrid)*** receive buffer, modifying the interrupt status register on the host, and raising the appropriate interrupt line. Upon receiving the interrupt, the host will read the ISR, clear it by writing back to the INIC's Interrupt Clear Register, and will then process its small receive buffer queue looking for receive buffers to be processed. Upon finding the small buffer, it will indicate the small amount of data up to the client to be processed by NetBIOS. It will also, if necessary, replenish the receive buffer pool on the INIC by passing off a pages worth of small buffers. Meanwhile, the NetBIOS client will allocate a memory pool ***(data buffers or re-assembly storage area)*** large enough to hold the entire NetBIOS message, and will pass this address or set of addresses down to the transport driver. The transport driver will allocate an INIC command buffer, fill it in with the list of addresses, set

the command type to tell the INIC that this is where to put the receive data, and then pass the command off to the INIC by writing to the command register. When the INIC receives the command buffer, it will DMA the remainder of the NetBIOS data, as it is received, into the memory address or addresses ***(data buffers or re-assembly storage area)*** designated by the host. Once the entire NetBIOS transaction is complete, the INIC will complete the command by writing to the response buffer with the appropriate status and command buffer identifier.

...

2.4.2. Slow-path receive

If the INIC receives a frame that does not contain a TCP segment for one of its CCB's, it simply passes it to the host as if it were a dumb NIC. ***If the frame fits into a small (hybrid) buffer (~200 bytes or less), then it simply fills in the small buffer with the data and notifies the host.*** Otherwise it places the data in a large buffer, writes the address of the large buffer into a small buffer, and again notifies the host. The host, having received the interrupt and found the completed small buffer, checks to see if the data is contained in the small buffer, and if not, locates the large buffer. Having found the data, the host will then pass the frame upstream to be processed by the standard protocol stack. It must also replenish the INIC's small and large receive buffer pool if necessary.

...

We take advantage of this fact by allocating large and small receive buffers. ***If a received frame fits in a small buffer, the INIC will use a small (hybrid) buffer. Otherwise it will use a large (data) buffer.*** A problem with that system then is preserving receive order. If we were to maintain a small and a large buffer queue, there would be no way to know in which order two frames, one small and one large, were received. A solution is to maintain a single receive queue of small buffers. We pass the small buffers in blocks of 16 at a time to the INIC, and they are guaranteed to be returned to us in the order in which they were given to the INIC. The small buffer contains status about the receive as well as small frames. If a received frame does not fit in the small buffer, then we allocate a large buffer and place a pointer to that large buffer in the small buffer. Thus, large buffers are only returned to the driver when attached to small buffers.

The remainder of this section covers this in greater detail.

3.2. Receive Interface

3.2.1. Receive Interface Overview

As mentioned above, the fast-path flow puts a ***header*** into a header ***(hybrid)*** buffer that is then forwarded to the host. The host uses the header to determine what further data is following, allocates the necessary host buffers ***(data buffers or re-assembly storage area)***, and these are passed back to the INIC via a command to the INIC. The INIC then fills these buffers from data it was accumulating on the card and notifies the host by sending a response to the command. ***Alternatively, the fast-path may***

> ***receive a header and data that is a complete request, but that is also too large for a header (hybrid) buffer. This results in a header and data buffer being passed to the host.*** This latter flow is identical to the slow-path flow which also puts all the data into the header ***(hybrid)*** buffer or, if the header buffer is too small, uses a large (2K) host ***(data)*** buffer for all the data.

For all the foregoing reasons, applicants respectfully assert that claims 4-7, 10-12, 14-16, 22, 26-27, 31 and 35-37 are enabled by the '296 app. and are entitled to the benefit of the '296 app.

IV. 35 U.S.C. §101

Claim 30 stands rejected under 35 U.S.C. §101 as allegedly being directed to non-statutory subject matter. In this regard, the Office Action states:

> Claim(s) 30 recites a "computer readable storage medium" which appear to cover both transitory and non-transitory embodiments. While Applicant's Specification may or may not provide examples of a medium as claimed, such examples do not explicitly define the term. The United States Patent and Trademark Office (USPTO) is required to give claims their broadest reasonable interpretation consistent with the specification during proceedings before the USPTO. *See In re Zletz*, 893 F.2d 31 9 (Fed. Cir. 1989) (during patent examination the pending claims must be interpreted as broadly as their terms reasonably allow). The broadest reasonable interpretation of a claim drawn to a medium as claimed typically covers forms of non-transitory tangible media **and** transitory propagating signals per se in view of the ordinary and customary meaning of the term, particularly when the specification is silent of an explicit definition. See MPEP 21 11.01. When the broadest reasonable interpretation of a claim covers a signal per se, the claim must be rejected under 35 U.S.C. § 101 as covering non-statutory subject matter. *See In re Nuijten*, 500 F.3d 1346, 1356-57 (Fed. Cir. 2007) (transitory embodiments are not directed to statutory subject matter) and *Interim Examination Instructions for Evaluating Subject Matter Eligibility Under 35 U.S.C. § 101*, Aug. 24, 2009; p. 2.
>
> The Examiner suggests that the Applicant add the limitation "non-transitory" to the medium as recited in the claim(s) in order to properly render the claim(s) in statutory form in view of their broadest reasonable interpretation in light of the originally filed specification.

Applicants have amended claim 30 as suggested by the Examiner, and respectfully assert that it is directed to statutory subject matter.

V. 35 U.S.C. §102

A. Muller

Claims 4-7, 10-12, 14-16, 22, 24, 26-27, 31 and 35-37 stand rejected under 35 U.S.C. 102(e) as allegedly being anticipated by Muller. As shown above, claims 4-7, 10-12, 14-16, 22, 26-27, 31and 35-37 are entitled to the benefit of the '296 app., which was filed well before Muller. Therefore, applicants respectfully assert that claims 4-7, 10-12, 14-16, 22, 26-27, 31 and 35-37 are not anticipated by Muller.

B. Born

Claims 1 and 30 stand rejected under 35 U.S.C. 102(e) as being anticipated by U.S. Patent No. 6,631,484 to Born ("Born"). Regarding claim 1, the Office Action states:

> Regarding claim 1, Born disclosed a method of storing a portion of a packet in a host computer memory, comprising:
>
> receiving a first packet at a communication interface (Born, col. 9, lines 37-39, Born disclosed monitoring for the presence of packets received by port 26/27);
>
> receiving a second packet at said communication interface (Born, col. 9, lines 37-39, Born disclosed monitoring for the presence of packets received by port 26/27);
>
> storing a header portion of said first packet in a hybrid storage area of a host computer (Born, Fig. 6, 88, col. 9, lines 39-42, Born disclosed placing the header in a FIFO);
>
> if said first packet includes a data portion, storing said data portion in a data storage area of said host computer (Born, Fig. 6, 91, col. 9, lines 45-46, Born disclosed if the packet is a large-size packet, placing the data portion in buffer 77,78, or 79); and
>
> if said second packet is smaller than a predetermined size, storing said second packet in said hybrid storage area (Born, Fig. 6, 90, col. 9, lines 46-69, Born disclosed if it is determined that the packet is a small-size packet, the packet's data is placed in the FIFO).

Applicants have amended claims 1 and 30 to recite, in part, "receiving from a network a first packet at a communication interface, said first packet including a Transmission Control Protocol (TCP) header." Born, on the other hand, is directed to an "interface apparatus provides a connection between a host having an IEEE 1394 input/output port and a mass storage device having an ATA input/output port." Abstract. In particular, Born teaches "a data processing system that is especially constructed and arranged to process the continuous asynchronous transmission and reception of relatively

small-size data packets while, at the same time, enabling the asynchronous transmission and reception of a half-duplex stream of relatively large-size data packets." Column 2, lines 11-16. "In order to accomplish two-way data transmit/receive functions in an IEEE 1394-to-ATA interface unit, a relatively small-size receive FIFO is associated with one data path, and a relatively small-size transmit FIFO is associated with a second data path." Column 2, lines 21-25. Neither "an IEEE 1394 input/output port" nor "an ATA input/output port" would be expected to receive a TCP header. Applicants respectfully submit that claims 1 and 30, as amended, are well removed from Born.

C. Thompson

Claims 17, 34 and 38 stand rejected under 35 U.S.C. 102(b) as being anticipated by U.S. Patent No. 5,917,828 to Thompson. Regarding claim 17, the Office Action states:

> Regarding claim 17, Thompson disclosed a method of network communication, the method comprising: providing a computer having a processor and a memory, the memory including first, second and third storage areas that are accessible by a communication interface (Fig. 11, memories 1108, 1122, and 1140);
>
> receiving, by the communication interface, a first packet, and storing the first packet in the first storage area (Thompson, Fig. 11, All packets received are first stored in cell FIFO 1108);
>
> receiving, by the communication interface, a second packet, storing a header of the second packet in the first storage area (Thompson, Fig. 11, All packets received are stored in cell FIFO 1108), and storing data of the second packet in the second storage area (Thompson, col. 8, lines 27-31, Thompson disclosed data from the received packets being stored in local memory buffer 1122 in certain conditions); and
>
> receiving, by the communication interface, a third packet, and storing data of the third packet in the third storage area, the third storage area containing data from a plurality of packets and corresponding to an application running on the computer, the third storage area containing no headers (Thompson, col. 8, lines 40-45, Thompson disclosed the contents of the local memory buffer, which include the payloads of received packets, to be stored in main memory buffer 1140).

Applicants have amended claim 17 to recite, in part, "receiving, by the communication interface, a third packet including data and a Transmission Control Protocol (TCP) header, and storing the data of the third packet in the third storage area,

the third storage area containing data from a plurality of packets and corresponding to an application running on the computer, the application denoted by the TCP header, the third storage area containing no TCP headers." Support for this amendment can be found, for example, in the '296 app. on page 21, line 3 - page 22, line 36.

Applicants respectfully assert that Thompson does not disclose this recitation. For example, Thompson discloses that the main memory buffer does include packet headers. For example, in column 7, lines 47-57, Thompson states:

> If the PDU is slightly larger than one memory buffer (4096+492 bytes), the status will be combined in local buffer memory 1006 with the pointer to the local memory buffer plus the remaining data bytes and this will be written as one burst to the status queues in main memory 1008. This is important because most large PDU transfers are equal to the memory page size plus a small TCP/IP or UDP/IP header and would normally use only a small amount of a second host memory buffer. This feature also combines the status write and the write of the overflow data into one larger burst write from local memory 1006 to host memory 1008.

In other words, when a TCP header is included in the received packet, the TCP header is transferred from the local memory buffer to the main memory buffer according to Thompson, in contrast to the recitation in claim 17 of "the third storage area containing no TCP headers."

For at least these reasons, applicants respectfully assert that Thompson does not anticipate claim 17 or any claim that depends from claim 17, such as claims 34 or 38.

VI. 35 U.S.C. §103

A. Claim 37

Claim 37 stands rejected under 35 U.S.C. 103(a) as being unpatentable over Born in view of U.S. Patent No. 5,870,394 to Oprea. In this regard, the Office Action states:

> Regarding claim 37, Born disclosed a communication interface configured to store packets for transfer to a host computer, comprising:
>
> a parser configured to determine whether a packet includes a data portion (Born, col. 9, lines 35-50, Born disclosed the function of placing a packet's data portion into a buffer which would require determining that there is a data portion); and
>
> a hybrid storage area (Fig. 6, 88, FIFO) configured to store:
>
> header portions of the plurality of packets (col. 8, lines 55-54); and

one or more packets smaller than a first predetermined size (col. 8 lines 46-50).

Born did not explicitly state having a re-assembly storage area configured to store data portions of a plurality of packets from a single communication flow.

In an analogous art, Oprea disclosed an apparatus for reassembly of data packets into messages in which the packets of a flow are stored in a re-assembly storage area (Fig. 2, 34).

One of ordinary skill in the art would have been motivated to use the reassembly function of Oprea within the teachings of Born since messages are generally made up of more than a single packet and as such, the packets must be reassembled at the receiving end in order to obtain the predictable result of communication to occur properly.

Applicants have amended claim 37 to recite "a parser configured to determine whether a packet includes a data portion and a Transmission Control Protocol (TCP) header; a re-assembly storage area configured to store data portions of a plurality of packets from a single communication flow; and a hybrid storage area configured to store: header portions of the plurality of packets, the header portions including TCP headers; and one or more packets smaller than a first predetermined size." Neither Born nor Oprea mentions TCP, and so applicants respectively assert that it would not have been obvious to modify Born to configure a parser to determine whether a packet includes a TCP header, and it also would not have been obvious to modify Born to store TCP headers in separate buffers from corresponding data but along with packets smaller than a predetermined size.

VII. Allowable Subject Matter

Applicants appreciate the indication that claims 2, 3, 8, 9, 18-21, 23, 28 and 29 contain patentable subject matter. As discussed above, however, applicants believe that all of the pending claims are allowable.

VIII. Conclusion

Applicants have responded to each item of the Office Action to demonstrate that the pending claims are in condition for allowance. In particular, applicants have shown that the pending claims have support in the'296 app. and a continuous chain of

applications up to and including the present, so that Muller is not prior art to any of the claims. Applicants appreciate the time and analysis the Examiner has invested in this application. Should the Examiner have any question about this response or this application, he is respectfully requested to telephone the undersigned.

Respectfully submitted,

/Mark Lauer/

Mark Lauer
Reg. No. 36,578
6601 Koll Center Parkway
Suite 245
Pleasanton, CA 94566
Tel: (925) 621-2121
Fax: (925) 621-2125